Available online at www.sciencedirect.com

ScienceDirect

St. Petersburg Polytechnical University Journal: Physics and Mathematics 1 (2015) 113–119

www.elsevier.com/locate/spjpm

Predicting the parameters of energy installations with laser ignition: Neural network models☆

Alexey A. Pastukhov

National Research University of Electronic Technology, 5 Pass. 4806, Zelenograd, Moscow, 124498, Russian Federation

Available online 30 July 2015

Abstract

This article considers the possibility of using artificial neural networks for predicting the parameters of the model energy installation with laser ignition. The main stages of creating a prognostic model based on an artificial neural network have been presented. Input data were analyzed by principal component method. The synthesized neural network was designed to predict the parameter value of the model in question. The artificial neural network was trained by a back-propagation algorithm. The efficiency of the artificial neural networks and their applicability to predicting parameter values of various rocket engine elements were demonstrated.

Keywords: Artificial neural network; Rocket engine; Laser ignition device; Prognostic model.

1. Introduction

Modern laser facilities are developing fast, and laser beam units are, due to their high emitting power and small weight, used in an increasingly diverse variety of areas of science and technology. For example, it is possible to use a laser device for propellant ignition. This application of laser would present an alternative to the conventional method of starting propellant combustion in rocket engines.

This so-called laser ignition device (LID) was designed by the Keldysh Research Centre (Moscow) together with the Energomash R&D complex (Khimki, Moscow oblast). Energomash has been bench-testing the LIDs since 2011 [1].

Developing and integrating new rocket engine elements such as LIDs is going to entail copious bench and *in situ* tests requiring much time and expense. Furthermore, the tested system itself is rather complex, with over 15 adjustable installation parameters, about 10 observables, and about a hundred accompanying measurements. Due to these difficulties, only 34 tests have been conducted from 2011 to 2013.

To save time and material costs, specifically, to reduce the number of *in situ* tests, it seems prudent to use mathematical methods and algorithms to predict the responses to inputs of the system as a whole and of the system's individual elements.

The goal of the present study is to design a trainable neural network for predicting the parameter values of a model energy installation with laser ignition.

☆ Peer review under responsibility of St. Petersburg Polytechnic University.

E-mail address: pastuhov1992@gmail.com.

http://dx.doi.org/10.1016/j.spjpm.2015.07.004

(Peer review under responsibility of St. Petersburg Polytechnic University).

2. The advantages of a trainable neural network

In the present paper, the predictive model of the system behavior has been constructed using the trainable neural network described in [2] that has the following advantages.

Firstly, each of the network's computing elements (neurons) is simple. A neuron is a weighted adder with the output defined as

$$y = f(u),\ \ u = \sum_{i=1}^{n} w_i x_i + w_0 x_0,$$

where x_i is an i-th input of a neuron, w_i is a synaptic weight of an i-th input connection, f is an activation function.

The input x_0 that is a fixed input signal $+1$ and its matching weight w_0 is called a neuron bias.

It is possible to configure each neuron and then construct compact mathematical models to solve prediction problems and uncover nonlinear functional connections. If there are a large number of connections between neurons, then, on one hand, this necessitates an increase in computations, but, on the other hand, it allows to fairly closely approximate functions of any complexity.

Secondly, the network is flexible. A neural network consists of neurons linked by weighted connections. There are a lot of types of neural networks differing in connection topology, neuron types, etc. It is also possible to construct new types of neural networks by, say, combining the existing types. This allows to solve diverse tasks, and, in particular, the prediction problem.

Thirdly, the model may be refined when new data arrives. A modular principle of training the artificial neural network implies that an input of new data does not cause the retraining of the whole system, but only individual neurons and connections are altered. This is important as training is a complex process and a network may lose its generalizing properties through total retraining.

In the present work the temperature of gas in the orifice plate of the model installation with laser ignition was chosen as the predicted parameter, as its value determines the result (outcome) of the whole experiment (success or failure). Besides, there is data for the chosen gas temperature for all tests (in no case did the sensor malfunction).

3. The stages of building a neural network prediction model

Let us list the stages of building this model.

1. Preparing the input data.
2. Analyzing the factor space and choosing the variables.
3. Preparing the factor space.
4. Designing the neural network model.
5. Training the neural network.

3.1. Preparing the input data

This stage largely defines the adequacy of the built model. The factor space including the measured parameters of the model installation is constructed from the data obtained in tests. This space is represented in relational form and divided into three groups.

The first group includes attributes and parameters whose values were known before the experiment. The purpose of forming this group is to provide input parameters for the model that allow to reliably distinguish between the conditions of different experiments. The first group includes four parameters, let us designate it as

$$A = \{A_1, A_2, A_3, A_4\},$$

where A_j are the vector columns of $A_j = (a_i^j)$, and i is the number of the experiment, $i = 1, 2, \ldots, n$ (n is the total number of experiments);

- $a_i^1 \in [15.18;\ 24.91]$ is the fuel temperature before the orifice plate (measured in degrees Celsius);
- $a_i^2 \in [74.11;\ 198.6]$ is the fuel pressure before the orifice plate (kgf/cm^2);
- $a_i^3 \in [0.1016;\ 2.15]$ is the oxidizer flow rate (kg/s);
- $a_i^4 \in [0.1065;\ 0.32]$ is the fuel flow rate (kg/s).

The second group includes the values of parameters being measured during the experiment that are not essential for obtaining the result (i.e. the temperature in the service compartments of the installation). These parameters provide input data redundancy and are used to configure and adjust the model. This group also consists of four parameters and we shall designate it as

$$B = \{B_1, B_2, B_3, B_4\},$$

where B_j are vector columns of $B_j = (b_i^j)$; i is the number of the experiment, $i = 1, 2, \ldots, n$ (n is the total number of experiments);

- $b_i^1 \in [0.325;\ 58.51]$ is the component flow rates ratio (kg/s);
- $b_i^2 \in [226.71;\ 322.6]$ is the pressure in the oxidizer container (kgf/cm^2);
- $b_i^3 \in [74.82;\ 203.14]$ is the pressure in the fuel container (kgf/cm^2);
- $b_i^4 \in [117.46;\ 161.66]$ is the pressure in the coolant containers (kgf/cm^2).

Let us call $X = A \cup B$ a set of input parameters. It can be given by a matrix

$$X = \begin{pmatrix} a_1^1 & \dots & a_1^k & b_1^1 & \dots & b_1^l \\ \dots & \dots & \dots & \dots & \dots & \dots \\ a_n^1 & \dots & a_n^k & b_n^1 & \dots & b_n^l \end{pmatrix},$$

where k is the number of the parameters of the first group of the factor space, and l is the number of parameters of the second group. The matrix has n rows and $m = k + l$ columns. In this case, $n = 29$, $m = 8$.

The third group of parameters includes the experimental values whose measurement determines the result. These values provide the output data in the experimental model. In this work we have chosen a single output parameter to predict: the gas temperature in the orifice plate. The group $Y = \{y_i\}$ will be named an output data set. Here i is the number of the experiment, $i = 1, 2, \dots, n$ (n is the total number of experiments); $y_i \in [-64, 02; 993, 52]$ is the gas temperature in the orifice plate, °C.

The output data set takes the form

$$Y = \begin{pmatrix} y_1 \\ \dots \\ y_n \end{pmatrix},$$

where $n = 29$.

Therefore, the set $V = X \cup Y$ is the factor space of the model.

3.2. Analyzing the factor space and choosing the variables

To design a neural network, we can transform the factor space so that it has less 'effective' attributes, ultimately leaving only the most information-rich data. This allows to cut down the number of input attributes of the factor space with minimum information loss, which, in turn, could simplify the architecture of the neural network. Additionally, cutting down the number of attributes allows to project the multi-dimensional factor space onto a space of lower dimension, which considerably facilitates analyzing it.

Another question is whether there is an optimal reversible linear transform of the output data space that makes it possible to reduce the dimension of this space. Simply cutting down a number of parameters of the factor space will cause the root-mean-square error of this transform to be equal to the sum of the variances of the excluded elements. Hence, the transform must have a small variance of its individual components.

The principal component analysis (called the Karhunen–Loeve transform in information theory) [2] allows to maximize the variance-reduction rate and thus maximize the probability of correctly choosing an acceptable parameter set.

Mathematically, the principal component method entails the spectral decomposition of a covariance matrix, which coincides with the singular value decomposition for the input data matrix [3].

The singular value decomposition for the input data matrix X is its representation as a product

$$X = S \cdot \Sigma \cdot C^T, \tag{1}$$

where S is an orthogonal matrix formed by the eigenvectors s_i of the matrix $X \cdot X^T$ corresponding to the eigenvalues λ_i, i.e. $X \cdot X^T s_i = \lambda_i s_i$; C is an orthogonal matrix formed by the eigenvectors c_i of the matrix $X^T \cdot X$ corresponding to the eigenvalues, i.e. $X^T \cdot X c_i = \lambda_i c_i$; Σ is a positive definite diagonal matrix with the elements

$$\sigma_1 \geq \dots \geq \sigma_i \geq \dots \geq \sigma_r \geq 0,$$

where $\sigma_i = \sqrt{\lambda_i}$; $i = 1, \dots, r = \mathrm{rank}(X)$. In this case $r = 8$.

The columns of the matrix S are called left-hand singular vectors. The columns of the matrix C are called right-hand singular vectors.

Going back to the decomposition (1) and taking into account that the elements of the matrix Σ are arranged in descending order, we find that the greatest contribution to matrix X is from the first columns of the matrix S and the first rows of the matrix C, as the greatest values of the matrix Σ correspond to them.

The relationship between the singular value decomposition and the principal component method is defined by the formulae

$$T = S \cdot \Sigma, \tag{2}$$

where T is the scores matrix containing the projections of the input data onto the subspace of principal components;

$$P = C, \tag{3}$$

where P is the loadings matrix that is the transfer matrix from the input space of variables X to the space of principal components.

There were only 29 experimental implementations, which is close to the lower limit for training neural networks with a large number of adjustable parameters, so it seems worthwhile to cut down the number of factor space parameters.

The factor space including the data of 29 tests is projected onto the space of principal components. The results are listed in Table 1 and Fig. 1. Evidently, the first

Table 1
The results of the contribution analysis of the eigenvalues obtained in 29 tests.

N	V	Total variance proportion, %	Cumulative eigenvalue	Cumulative variance proportion, %
1	3.324898	41.56122	3.324898	41.5612
2	1.906212	23.82764	5.231109	65.3889
3	1.258792	15.73490	6.489902	81.1238
4	0.978690	12.23363	7.468592	93.3574
5	0.344825	4.31031	7.813417	97.6677
6	0.183848	2.29810	7.997265	99.9658
7	0.002400	0.03000	7.999666	99.9958
8	0.000334	0.00418	8.000000	100.0000

Notations: N is the number of eigenvalues (matches the component number), V is an eigenvalue.

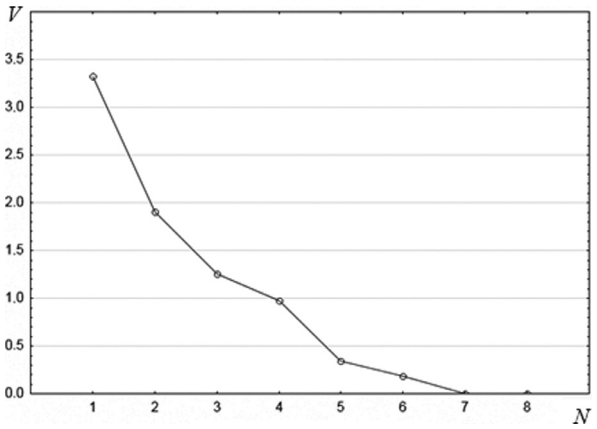

Fig. 1. The results of projecting the factor space onto the space of principal components: N is the number of eigenvalues, V is an eigenvalue.

three principal components account for 81.2% of the total variance. Components 7 and 8 make practically no contribution to the variance. Over 90% of the total variance is provided by the first four principal components, but in this case, the goal of applying the method is to reduce the number of the factor space attributes as much as possible.

Therefore, it seems appropriate to focus only on the first three principal components.

We shall analyze the factor load matrix (Table 2) to understand what factors should be kept in the factor space. It is apparent that the first factor strongly correlates with elements 3 and 5 of the input data matrix X, the second and the third ones correlate with elements 2 and 4 respectively (the values are highlighted in bold).

It should be noted here that variable 5 (the ratio of the component flow rates) correlates rather strongly with elements 3 and 4. This can be seen from the factor space correlation matrix in Table 3.

Table 2
Factor load matrix for the three principal components.

	F_1	F_2	F_3
X_1	0.360048	0.044197	0.161626
X_2	0.655228	**0.703099**	–0.227213
X_3	**–0.841152**	0.440810	–0.257437
X_4	0.558408	–0.181547	**–0.713014**
X_5	**–0.839159**	0.424273	–0.241175
X_6	–0.597731	–0.345838	–0.633068
X_7	0.661975	0.694228	–0.242636
X_8	0.496894	–0.633303	–0.297604

Notations: $X_1, X_2, \ldots, X_8$ are the columns of the input parameter set X; F_1, F_2, F_3 are, respectively, the first, the second, and the third principal components.

This result is unsurprising, as variable 5 is the ratio of variables 3–4.

Thus, three variables are included into the output factor space for the model: 2, 3, and 4. Variable 5 is excluded, as it is equal to the ratio of variable 3 to variable 4.

3.3. Preparing the factor space

Before using the factor space as data for designing the neural network, we must prepare it by solving the problem of neural saturation. Once the saturation is reached, the values calculated by neurons, start approaching the domain boundary of the activation function, which causes a slowdown in the training process and to network paralysis.

The data used for training the neural network is normed so that the neuron input does not fall into the domain of the small derivative of the activation function [4], for which the sigmoid function is used. In this case the input parameters take both positive and negative values, so a hyperbolic tangent

$$f(x) = \mathrm{th}(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

was chosen as the activation function.

This function is continuously differentiable on the whole real axis:

$$f'(x) = 1 - \mathrm{th}^2(x),$$

and allows to solve the problem of noise saturation, which is necessary for the training algorithms to properly function.

Furthermore, the desired neural response of the output layer must be biased inwards by a certain amount from the boundary of the domain of the activation function. This is achieved by initializing the neuron bias by nonzero values.

Table 3
Factor space correlation matrix.

	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8
X_1	1.000000	0.149644	–0.231507	0.204460	–0.301840	–0.173092	0.153268	–0.056537
X_2	0.149644	1.000000	–0.198955	0.342744	–0.198955	**–0.483203**	**0.999401**	–0.001569
X_3	–0.231507	–0.198955	1.000000	–0.342020	**0.983270**	**0.488857**	–0.203656	**–0.601238**
X_4	0.204460	0.342744	–0.342020	1.000000	**–0.405599**	0.160197	0.364045	**0.456364**
X_5	–0.301840	–0.198955	**0.983270**	**–0.405599**	1.000000	**0.463971**	–0.198496	**–0.511579**
X_6	–0.173092	**–0.483203**	**0.488857**	0.160197	**0.463971**	1.000000	**–0.475425**	0.030348
X_7	0.153268	**0.999401**	–0.203656	0.364045	–0.198496	**–0.475425**	1.000000	0.007380
X_8	–0.056537	–0.01569	**–0.601238**	**0.456364**	**–0.511579**	0.030348	0.007380	1.000000

Notations: $X_1, X_2, \ldots, X_8$ are the columns of the input parameter set X.
Values greater than 0.4 in absolute value are in bold.

The last stage of preparing the data includes pre-initializing the free parameters (weights) of the neural network. The initialization was done using the Nguyen–Widrow method [5].

Then we can proceed with designing the neural network model itself.

3.4. Designing a neural network model

Ref. [6] analyzed several types of neural networks in view of their application to solving various problems. Two types of networks have been deemed the most appropriate for solving prediction problems: a multilayer perceptron (MLP) and a radial basis function (RBF). Generally, RBF networks produce a more accurate result but require a lot of experimental data. When the dimension of the training sample is relatively small, as is the case for this study, it is preferable to use MLP networks.

The main problem with designing a neural network model is finding the number of hidden-layer neurons.

The number of connections in a network is estimated by the formula following from the Arnold–Kolmogorov–Hecht-Nilsen theorems [7]:

$$\frac{mN}{1+\log_2 N} \le L_w \le m\left(\frac{N}{m}+1\right)(n+m+1)+m, \quad (4)$$

where n, m are the dimensions of the input and output signals, respectively, N is the number of elements in the training sample; L_w is the total number of connections in the neural network.

If we take into account the fact that all neurons of each layer are connected to all neurons of the adjacent layer, and there are no connections between the same-layer neurons, and also keep in mind Eq. (4), then the number of hidden-layer neurons is defined by the following formulae [8]:

$$L = \frac{L_w}{n+m}; \quad (5)$$

$$\frac{22}{1+\log_2 22} \le L_w \le 116; \quad (6)$$

$$\frac{22}{1+\log_2 22} \approx 4. \quad (7)$$

Thus, the number of network connections L_w lies in the range $4 \le L_w \le 116$. Using the range (6) and Eq. (5) we obtain that the number L of the hidden neurons of the network falls in the range $1 \le L \le 29$ (rounded off to the nearest integer).

As noted in Ref. [2], it is sufficient for a proper generalization that the size of the training set would satisfy the relation

$$N = O\left(\frac{W}{\varepsilon}\right), \quad (8)$$

where W is the total number of free parameters, ε is a tolerable accuracy for the classification error, N is the size of the training set.

Unfortunately, as previously stated, our study deals with only 29 experimental implementations. So to obtain the tolerable value of no more than $\varepsilon \approx 0.35$ for the three input parameters, the number of hidden-layer neurons must be no more than two. Thus, let us accept the number of hidden-layer neurons L to equal 2. The topology of the neural network obtained is shown in Fig. 2.

After forming the topology of the neural network we can move on to the last stage of constructing the neural network prediction model that is the training.

3.5. Training the neural network

Mathematically, the task of training the neural network means finding the functional dependence

$$F : F(X_i) \to y_i, \quad i = 1, 2, \ldots, 29,$$

where X_i is the i-th column of the input parameter matrix X; y_i is the i-th row of the output parameter matrix (in this case the i-th element of the vector column) Y.

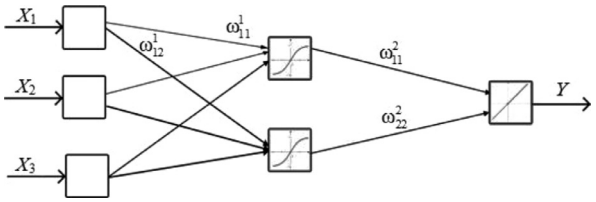

Fig. 2. Topology of a neural network to predict the gas temperature in the orifice plate of an installation with laser ignition; X_i is the i-th network input, Y is the network output, ω_{ij}^k is the connection weight of the i-th neuron of the k-th layer with the j-th neuron of the $(k+1)$-th layer.

This is achieved by minimizing the objective function of the neural network error, calculated by the least square method:

$$E(\{\omega_{ij}\}) = \frac{1}{2}\sum_{k=1}^{n}(r_k - y_k)^2, \tag{9}$$

where r_k is the value of the k-th output of the neural network, y_k is the objective value of the k-th output, n is the number of neurons in the output layer, ω_{ij} is the weight of the connection between the neurons i and j.

4. Predicting the gas temperature in the orifice plate

Out of 29 available data sets obtained through testing the LID, we used 22 to create a predictive model, 3 to independently test the network, and the remaining 4 to validate the training.

Each set includes the pair of an input vector (a column from the input parameter matrix X) and an output value (a corresponding value from the output parameter vector Y).

The network was trained using the error back-propagation algorithm that is a modified gradient descent method [2]; weighting coefficient increments for each iteration were found by the formula

$$\Delta\omega_{ij} = -\eta\frac{dE}{d\omega_{ij}}, \tag{10}$$

where ω_{ij} is the weighting coefficient of the connection between the i-th neuron of layer $k-1$ with the j-th neuron of layer k; η is a training rate coefficient, $0 < \eta < 1$.

According to the algorithm, the training procedure stops either when the error of the given value is attained, or 50 training epochs have been completed.

Fig. 3 shows regressograms of the training the neural network, plotted using the NNtool box from the MATLAB package.

Table 4 lists the results of analyzing plots 2 in Fig. 3 for a synthesized neural network, including the ratios of actual and target values.

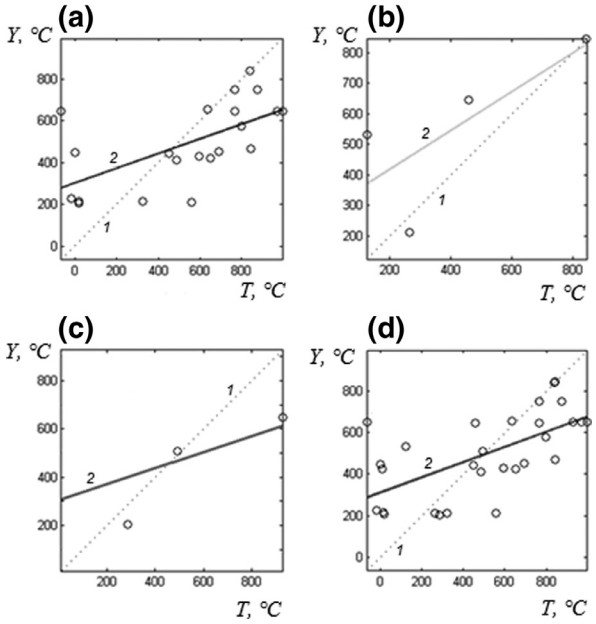

Fig. 3. Regressograms of the training the neural network (plots of the actual values Y versus target values T): training results (a), retraining tests (b) and checking the test set (c), and the total result (d); 1 is the straight line $Y = T$, 2 is the straight line going through the center of the data cloud; the symbols are the test data.

Table 4
The results of training the neural network.

Plot in Fig. 3	R	$Y(T)$, °C
(a)	0.62102	$0.35T + 3.0 \times 10^2$
(b)	0.75051	$0.64T + 2.9 \times 10^2$
(c)	0.68722	$0.33T + 3.0 \times 10^2$
(d)	0.62025	$0.36T + 3.1 \times 10^2$

Notations: R is the neural network output/target value ratio, $Y(T)$ is the approximated linear dependence of actual values on the target values T.

For $R = 1$ there is an exact linear relationship between the neural network output and the target value, while if R is close to zero, there is no linear relationship between these values [9].

The results obtained for the test set (the three experimental implementations that were not used in training) can be seen in Fig. 4 and Table 5.

5. Conclusion

We can confirm that the obtained model is not particularly suitable for accurately predicting the parameter value, since, as expected, the relative deviations of the values of the test and the training sets from the target values averaged about 35%. This value is so high because there are few training examples.

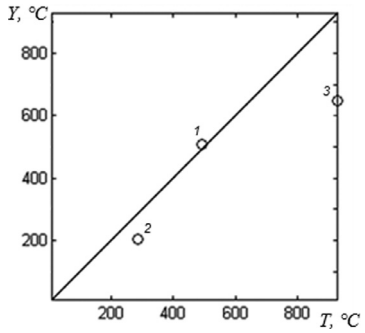

Fig. 4. Graphical estimate of the temperature prediction results for the test set: T is the target value, Y is the actual value predicted by the network, the straight line is the trend; points *1–3* are the elements of the test set.

Table 5
Checking the gas temperature predictions for the implementations not used for neural network training.

Point in Fig. 4	Value, °C		Deviation from the value	
	T	Y	$T-Y$, °C	$2\times10^2\,\lvert T-Y\rvert/(T+Y)$, %
1	506.3	492.7	13.6	3
2	285.6	202.3	83.3	34
3	651.2	929.3	–278.1	35

The goal of this study, however, was not to create an accurate model (which would be impossible with so few experimental implementations), but to merely demonstrate that the artificial neural network methods are applicable to this subject area.

We should specifically stress that it is possible to broaden the factor space and, as a result, improve the constructed model, with more *in situ* tests (including failed ones).

Thus, the present work has examined designing a neural network prediction model based on experimental data that can uncover functional connections between parameters without any history available. The designed model allows to predict the output parameter matching the results of testing a model installation with laser ignition with a predetermined accuracy.

The results obtained prove that it is conceptually possible to create prediction models for the components and units of tested rocket engines. Models may be created using the means of artificial neural networks and the rather wide data base of conducted tests.

References

[1] A. Khromtsova, Lasers ignite the engines, Ross. Cosmos. 8 (92) (2013) 2–4.
[2] S. Khajkin, Nejronnye seti: polnyj kurs [Neural Networks: A Complete Course,], 2nd ed., ID Vil'yams, Moscow, 2008.
[3] S.A. Ajvazyan, V.M. Bukhshtaber, I.S. Enyukov, L.D. Meshalkin, Prikladnaya statistika. Klassifikatsiya i snizhenie razmernosti [Applied Statistics. Classification and Dimension Reduction], Finansy i statistika, Moscow, 1989.
[4] V.S. Medvedev, V.G. Potemkin, Nejronnye seti [Neural Networks], Dialog MIFI, Moscow, 2001.
[5] Y. LeCun, Efficient Learning and Second-order Methods, MIT Press, Cambridge, MA, 1993.
[6] P.E. Rodionov, Metodika izvlecheniya znanij v zadachakh analiza ryadov dinamiki s ispol'zovaniem nejronnykh setej: dis. … kand. tekhn. nauk: 05.13.17 [Methods of Extraction of Knowledge for the Analysis of Time Series Using Neural Networks], Moscow, 2003.
[7] R. Hecht-Nielsen, Counterpropagation networks, Appl. Optics 26 (23) (1987) 4979–4984.
[8] D.V. Devyatykh, O.M. Gerget, I.V. Mikhalenko, Primenenie iskusstvennykh nejronnykh setej dlya prognozirovaniya razvitiya perinatal'nogo porazheniya nervnoj sistemy [Artificial neural network implementation for newborn nervous system perinatal damage diagnosing], Izvestiya VolgGTU 16 (8) (2013) 77–80 (111).
[9] M.H. Beale, M.T. Hagan, H.B. Demuth, Neural Network Toolbox User's Guide, MathWorks, Inc., 2014, pp. 77–78, available at: http://www.mathworks.com/help/pdf_doc/nnet/nnet_ug.pdf